

# Trabalho Prático

## Programação Funcional

Ano Letivo 2025 – 2026

Ver 1.0

### Tema do Trabalho

Pretende-se desenvolver um conjunto de funções e uma aplicação para gestão de torneios desportivos, permitindo o emparelhamento dos atletas ou equipas, o registo dos resultados e o apuramento das classificações finais.

Um exemplo seria um torneio de bilhar, em que um conjunto de jogadores é sucessivamente emparelhado para jogar pelo método AVE (às vezes referido como sistema AVE ou sistema AvePool) e apurar um vencedor do torneio. Outro exemplo seria um torneio de andebol, em que um conjunto de equipas são emparelhadas pelo método de Eliminatórias Diretas e jogam entre si para apurar uma vencedora do torneio.

### Os métodos de Emparelhamento

Os métodos de emparelhamento a implementar são dois: AVE e Eliminatórias Diretas.

#### Método AVE

O AVE (Average System ou Average de Vitórias e Encontros) é um sistema de emparelhamento progressivo baseado na média de resultados obtidos por cada jogador ao longo do torneio. Em vez de eliminar jogadores após uma derrota (como no sistema de eliminação simples), o método atribui pontuações médias e emparelha jogadores com desempenho semelhante em cada ronda.

Objetivos do sistema AVE:

- Promover equilíbrio entre os emparelhamentos (evitando confrontos demasiado desequilibrados);
- Garantir justiça estatística, já que todos jogam várias rondas antes de apurar os melhores;
- Permitir torneios rápidos, sem necessidade de longas fases eliminatórias;
- Minimizar o impacto do azar num único jogo (um mau início não elimina o jogador).

### Funcionamento passo a passo

1. Ronda inicial:
  - Todos os jogadores são emparelhados de forma aleatória.
  - Cada jogo tem um resultado final (vitória/derrota e número de partidas ganhas/perdidas).
2. Atribuição de pontos:
  - Cada jogador recebe um “AVE” (average), que corresponde à razão entre o número de partidas ganhas e o número total de partidas disputadas.
  - Exemplo: se um jogador ganha 5 partidas e perde 3, o seu AVE é  $5 / (5 + 3) = 0,625$ .
3. Emparelhamento nas rondas seguintes:
  - Os jogadores são ordenados pelo valor do AVE.
  - São emparelhados jogadores com AVE semelhantes, de modo a manter jogos equilibrados.
  - Evitam-se, sempre que possível, repetições de confrontos.
4. Atualização contínua:
  - A cada ronda, o AVE é atualizado com os novos resultados.
  - O sistema ajusta-se dinamicamente — quem tem melhor desempenho enfrenta adversários mais fortes.
5. Classificação final:
  - No final do número predefinido de rondas (ou quando restam poucos jogadores invictos), a classificação final é feita com base:
    - No valor médio AVE (principal critério);
    - No número total de vitórias;
    - E, se necessário, em critérios de desempate como o confronto direto ou a diferença total de frames/partidas ganhas.

### Exemplo simples

#### Jogador Jogos Ganhos Jogos Perdidos AVE

A	6	2	0.75
B	5	3	0.625
C	3	5	0.375
D	2	6	0.25

Na ronda seguinte, **A joga com B** (valores próximos), e **C joga com D**.

No Anexo I – Exemplo de um torneio com sistema AVE, é apresentado um exemplo completo de um torneio com 8 jogadores com 3 rondas em sistema AVE.

Ficheiros de dados:

Os ficheiros “torneio\_ave\_vila\_real.txt” e “resultados\_torneio\_ave\_vila\_real.txt” contém os dados de um torneio, incluindo: a designação do torneio, o número de rondas, os resultados dos jogos, a classificação dos jogadores (número de vitórias, derrotas e AVE).

Os ficheiros CSV “torneio\_ave\_vila\_real.csv” e “resultados\_torneio\_ave\_vila\_real.csv” contém a mesma informação que os respetivos TXT.

Para processamento deve usar os CSV.

## Método de Eliminatórias Diretas

O método de emparelhamento de eliminatórias diretas (também conhecido como sistema de eliminação simples ou “knockout”) é um dos formatos mais comuns para a gestão de competições desportivas. É amplamente utilizado em torneios de modalidades como ténis, bilhar, futebol, artes marciais, xadrez, entre outros.

Neste método, os jogadores ou equipas são emparelhados em confrontos diretos, em que o vencedor avança para a ronda seguinte e o perdedor é eliminado da competição. O processo repete-se até restar apenas um participante, que é declarado campeão.

### Funcionamento passo a passo

1. Número de participantes
  - Idealmente, o número de participantes deve ser uma potência de 2 (ex.: 8, 16, 32, 64), pois assim o torneio decorre sem rondas intermédias nem “byes”.
  - Se o número de jogadores não for potência de 2, atribuem-se “byes” (passagens automáticas à ronda seguinte) a alguns participantes — normalmente aos cabeças-de-série — para equilibrar o quadro.
2. Emparelhamento inicial
  - Pode ser feito:
    - Por sorteio (aleatório);
    - Por ranking ou cabeça-de-série (de forma a evitar que os melhores se encontrem nas primeiras rondas);
    - Por regiões ou grupos (em torneios geograficamente organizados).
3. Progressão
  - Cada ronda reduz o número de competidores a metade.
  - Exemplo:  
16 → 8 → 4 → 2 → 1 (campeão).
4. Duração
  - O número total de rondas é dado por:  
$$N^{\circ} \text{ de rondas} = \log_2(n)$$

onde  $n$  é o número de participantes.

Neste trabalho assume-se que o número de equipas é potência de 2 e o emparelhamento é sempre por sorteio.

### Exemplo simples

Exemplo prático com 8 participantes.

Ronda	Jogo Emparelhamento Vencedor		
1ª	J1	A vs H	A
	J2	B vs G	B
	J3	C vs F	F
	J4	D vs E	D
2ª (Meias-finais)	J5	A vs B	A
	J6	F vs D	D
Final	J7	A vs D	<b>A (Campeão)</b>

Ficheiros de dados:

Os ficheiros “torneio\_16\_clubes.txt” e “resultados\_torneio\_16\_clubes.txt” contém informação com: a designação do torneio, a lista de equipas, e o emparelhamento de acordo com o método de Eliminatórias Diretas, incluindo a identificação dos jogos.

Os ficheiros “torneio\_16\_clubes.csv” e “resultados\_torneio\_16\_clubes.csv” contém a mesma informação que os respetivos TXT.

Para processamento deve usar os CSV.

## Tarefas

### T1 – Ler os ficheiros csv, criar uma estrutura de dados e imprimir a informação no ecrã

T1.1 - Deve criar as seguintes funções de leitura, que recebem como parâmetro o nome do ficheiro, abrem e leem o ficheiro, devolvendo ação de IO e uma estrutura de dados com o respetivo conteúdo.

Deve usar tipos de dados simples (inteiros, decimais, caracteres, booleanos e strings) e compostos (tuplas e listas).

```
readTorneioAVE :: string -> IO estrutura_de_dados_torneio_ave

readResultadosTorneioAVE :: string -> IO estrutura_de_dados_resultados_ave

-- usar os ficheiros “torneio_ave_vila_real.csv” e
“resultados_torneio_ave_vila_real.csv”

readTorneioElim :: string -> IO estrutura_de_dados_torneio_elim

readResultadosTorneioElim :: string -> IO estrutura_de_dados_resultados_elim

-- usar os ficheiros “torneio_16_clubes.csv” e “resultados_torneio_16_clubes.csv”

-- em que a string representa o nome do ficheiro e estrutura_de_dados a respetiva
estrutura de dados resultante.
```

T1.2 - Deve criar as seguintes funções de escrita no ecrã, que recebem como parâmetro a estrutura de dados previamente definida e devolvem uma ação de IO, imprimindo no ecrã a respetiva informação num formato semelhante aos respetivos ficheiros TXT.

```
printTorneioAVE :: estrutura_de_dados_torneio_ave -> IO ()

printResultadosTorneioAVE :: estrutura_de_dados_resultados_ave -> IO ()

printTorneioElim :: string -> estrutura_de_dados_torneio_elim -> IO ()

printResultadosTorneioElim :: estrutura_de_dados_resultados_elim -> IO ()
```

ENTREGÁVEIS: modulo fileread, no ficheiro fileread.hs; relatório no ficheiro “PL#G#Tarefa1.pdf”. PL# identifica a turma e G# o número do grupo.

## T2 – Emparelhamento com AVE e Eliminação Direta

T2.1 – Deve criar uma função que recebendo duas estruturas de dados, lidas por `readTorneioAVE` e `readResultadosTorneioAVE`, com informação do torneio, dos participantes e das rondas já executadas, devolva uma estrutura de dados, adicionando uma nova ronda seguinte.

```
runAVEparing:: estrutura_de_dados_torneio_ave -> estrutura_de_dados_resultados_ave -> (estrutura_de_dados_resultados_ave)
```

T2.2 – Deve criar uma função que recebendo a estruturas de dados, lida por `readTorneioElim`, com informação do torneio e dos participantes, devolva uma estrutura de dados com o emparelhamento de todas as rondas.

```
runElimParing:: estrutura_de_dados_torneio_elim -> estrutura_de_dados_resultados_elim
```

ENTREGÁVEIS: modulo pairing, no ficheiro pairing.hs; relatório no ficheiro “PL#G# Tarefa2.pdf”. PL# identifica a turma e G# o número do grupo.

## T3 – Atualizar da informação após cada jogo

T3.1 - Deve criar uma função pura que recebe o resultado de um jogo e as estrutura de dados `estrutura_de_dados_torneio_ave` e `estrutura_de_dados_resultados_ave` e devolver uma dupla com as estruturas de dados atualizadas.

```
updateAVE :: dados_de_jogo -> estrutura_de_dados_torneio_ave -> estrutura_de_dados_resultados_ave -> (estrutura_de_dados_torneio_ave, estrutura_de_dados_resultados_ave)
```

T3.1 - Deve criar uma função pura que recebe o resultado de um jogo e as estrutura de dados `estrutura_de_dados_torneio_elim` e `estrutura_de_dados_resultados_elim` e devolver uma dupla com as estruturas de dados atualizadas.

```
updateElim :: dados_de_jogo -> estrutura_de_dados_torneio_elim -> estrutura_de_dados_resultados_elim -> (estrutura_de_dados_torneio_elim, estrutura_de_dados_resultados_elim)
```

ENTREGÁVEIS: modulo update, no ficheiro update.hs; relatório no ficheiro “PL#G# Tarefa3.pdf”. PL# identifica a turma e G# o número do grupo.

## T4 -Gravação em ficheiros csv

Deve criar uma função que recebe uma string, correspondendo ao nome de um ficheiro, e a uma estrutura de dados, conforme anteriormente descritas, e gravar os dados em ficheiro em formato csv, conforme os ficheiros de exemplo.

T4.1 – Para as estruturas de dados `estrutura_de_dados_torneio_ave` e `estrutura_de_dados_resultados_ave`

```
saveTorneioAVE :: string -> estrutura_de_dados_torneio_ave -> IO ()
```

```
saveResultadosTorneioAVE :: string -> estrutura_de_dados_resultados_ave -> IO ()
```

T4.2 – Para as estruturas de dados `estrutura_de_dados_torneio_elim` e `estrutura_de_dados_resultados_elim`

```
saveTorneioElim :: string -> estrutura_de_dados_torneio_elim -> IO ()
```

```
saveResultadosTorneioElim :: string->estrutura_de_dados_resultados_elim -> IO ()
```

ENTREGÁVEIS: modulo `filesave`, no ficheiro `pairing.hs`; relatório no ficheiro “PL#G# Tarefa4.pdf”. PL# identifica a turma e G# o número do grupo.

### T5 – Programa para gestão de torneios com emparelhamento AVE ou Eliminação Direta

Deve criar uma aplicação informática com as seguintes funcionalidades:

- Permita criar um torneio, com designação, tipo de emparelhamento e outra informação acessória. A identificação dos participantes deve ser lida de um ficheiro csv, com formato semelhante a “torneio\_ave\_vila\_real.csv” ou “torneio\_16\_clubes.csv”.
- Execute o respetivo emparelhamento, seguindo o método escolhido pelo utilizador.
- Permita fazer a atualização da informação com resultados de novos jogos durante o decurso do torneio.
- Permita apurar os vencedores (1º, 2º e 3º classificados).
- Permita visualizar a informação do torneio no ecrã

ENTREGÁVEIS: modulo `application`, no ficheiro `application.hs`; relatório no ficheiro “PL#G# Tarefa5.pdf”. PL# identifica a turma e G# o número do grupo.

## Requisitos do Relatório

O relatório deve ter uma página A4 e indicar claramente para cada tarefa:

- Qual o contributo de cada elemento do grupo e o tempo despendido.
- Justificar a organização do código
- Justificar as variáveis e valores.

O código deve ser comentado, descrevendo, para cada função, o objetivo da função, os parâmetros que recebe e o resultado que devolve. Ver o modelo no Anexo II.

## CrITÉrios de Avaliação

O trabalho prático é realizado ao longo do semestre e os entregáveis devem ser entregues no infostudante de acordo com os prazos definidos no calendário de entrega. Não são admitidos trabalhos fora de prazo.

A avaliação segue a seguinte formula:

$$NF = 0,1*T1 + 0,1*T2 + 0,1*T3 + 0,1*T4 + 0,1*T5 + 0,5*APRES$$

Em que: NF é Nota Final do trabalho prático; T1, T2, T3, T4 e T5 são as classificações dos entregáveis das diversas Tarefas; e APRES é a classificação da apresentação final.

Além da submissão online, nas aulas seguintes, o professor da PL poderá pedir aos grupos uma pequena explicação sobre o material entregue. No final do semestre haverá uma sessão de apresentação do trabalho.

Cada elemento do grupo deve ser capaz de explicar qual foi o seu contributo, porque tomou as opções apresentadas no trabalho, e o que faria se as premissas específicas do enunciado fossem modificadas.

## Prazos

Os entregáveis devem ser submetidos até às 23h50 dos dias a seguir indicados. Não são aceites submissões fora de prazo.

- Tarefa 1, 26 de outubro
- Tarefa 2, 9 de novembro
- Tarefa 3, 23 de novembro
- Tarefa 4, 7 de dezembro
- Tarefa 5, 4 de janeiro
- A apresentação final decorrerá na aula prática da semana de 5 de janeiro.

## Observações

- O trabalho pode ser realizado em grupos de até 3 elementos, conforme indicação do docente.
- É obrigatória a originalidade do trabalho. Plágio resultará em penalização severa, de acordo com o regulamento académico da instituição.
- Dúvidas ou esclarecimentos devem ser encaminhados ao docente responsável pela unidade curricular.

## Anexo I – Exemplo de um torneio com sistema AVE

Exemplo completo (8 jogadores, “corrida a 5” frames por encontro) com 3 rondas em sistema **AVE**. Em cada ronda, os emparelhamentos são feitos por **AVE semelhante** evitando repetições.

### Ronda 1 (emparelhamento inicial aleatório)

- A vs B → **A 5–3 B**
- C vs D → **C 2–5 D**
- E vs F → **E 5–1 F**
- G vs H → **G 4–5 H**

### AVE após R1 (frames ganhos / total):

- E:  $5/6 = 0,833$
- D:  $5/7 = 0,714$
- A:  $5/8 = 0,625$
- H:  $5/9 = 0,556$
- G:  $4/9 = 0,444$
- B:  $3/8 = 0,375$
- C:  $2/7 = 0,286$
- F:  $1/6 = 0,167$

### Ronda 2 (ordenar por AVE e emparelhar vizinhos; evitar repetição)

Ordenação: E, D, A, H, G, B, C, F

Emparelhamentos: **E–D, A–H, G–B, C–F**

Resultados:

- E vs D → **E 4–5 D**
- A vs H → **A 5–4 H**
- G vs B → **G 5–3 B**
- C vs F → **C 3–5 F**

### AVE acumulado após R2

- D:  $10/16 = 0,625$
- E:  $9/15 = 0,600$
- A:  $10/17 = 0,588$
- G:  $9/17 = 0,529$
- H:  $9/18 = 0,500$

- F: 6/14 = **0,429**
- B: 6/16 = **0,375**
- C: 5/15 = **0,333**

**Ronda 3 (evitar repetir E-D; escolher vizinhos mais próximos)**

Ordenação: D, E, A, G, H, F, B, C

Emparelhamentos: **D-A, E-G, H-F, B-C**

Resultados:

- D vs A → **D 4-5 A**
- E vs G → **E 5-3 G**
- H vs F → **H 5-2 F**
- B vs C → **B 4-5 C**

**Classificação final (por AVE; depois vitórias totais/critérios de desempate)**

Posição	Jogador	Frames Ganhos	Frames Perdidos	Total	AVE
1	<b>E</b>	14	9	23	<b>0,609</b>
2	<b>A</b>	15	11	26	<b>0,577</b>
3	<b>D</b>	14	11	25	<b>0,560</b>
4	<b>H</b>	14	11	25	<b>0,560</b>
5	<b>G</b>	12	13	25	<b>0,480</b>
6	<b>C</b>	10	14	24	<b>0,417</b>
7	<b>B</b>	10	15	25	<b>0,400</b>
8	<b>F</b>	8	13	21	<b>0,381</b>

Nota sobre desempates: D e H empatam no AVE (0,560) e em vitórias de encontro (2-1).  
Um torneio real pode usar **diferença de frames, confronto direto** (se jogaram), ou **força do adversário** (soma dos AVEs dos oponentes) para ordenar.

## Anexo II – Modelo do relatório

Deve manter os títulos e substituir o texto em itálico.

----

### Programação Funcional

Ano letivo 2025-2026

#### **Grupo PL e número do grupo**

*número e nome, por ordem alfabética;*

*número e nome, por ordem alfabética;*

*número e nome, por ordem alfabética;*

#### **Contributos**

*número e nome*

*Tempo despendido em horas*

*Descrição do contributo*

*número e nome*

*Tempo despendido em horas*

*Descrição do contributo*

*número e nome*

*Tempo despendido em horas*

*Descrição do contributo*

#### **Relatório do trabalho efetuado**

*Justificar as opções de organização do código*

*Explicar o propósito das funções, seus parâmetros e valores retornado.*